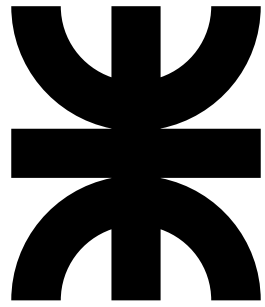
# Informática II
## Puerto serie

Gonzalo F. Perez Paina



Universidad Tecnológica Nacional
Facultad Regional Córdoba
UTN-FRC

– 2019 –

# Introducción

## Comunicación serie

- Necesita conversor paralelo/serie para “serializar/des-serializar” los datos
- Para la comunicación serie RS232 se utilizan las UARTs (Universal Asynchronous Reveiver-Transmitter)

# Introducción

## Comunicación serie

- Necesita conversor paralelo/serie para "serializar/des-serializar" los datos
- Para la comunicación serie RS232 se utilizan las UARTs (Universal Asynchronous Reveiver-Transmitter)

Ventajas de la transmisión de datos serie (vs. paralelo):

- En la comunicación en paralelo se necesitan tantos cables como ancho de la palabra de comunicación
- Los cables pueden tener mayor longitud (RS232)
- Es posible reemplazar la conexión cableada por comunicación inalámbrica, como p.e. utilizando IR, láser, etc.
- Los microcontroladores en general cuentan con puertos de comunicación serie (UART, $I^2C$, SPI, etc.)

# Hardware

Las especificaciones eléctricas del puerto serie se definen en el estándar EIA[1], que para el RS232C son

- Un *espacio* (0 lógico) entre +3 y +25V
- Una *marca* (1 lógico) entre -3 y -25V
- Los valores entre +3V y -3V representan estados indefinidos
- Corriente de cortocircuito <500mA

(entre otros parámetros como: capacitancia máxima de línea, baudrate máximo, etc.)

[1] EIA: Electronics Industry Association - RS: Recommended Standard EIA-232C (1969), EIA-232D (1986), EIA-232 (1991)

# Hardware

Las especificaciones eléctricas del puerto serie se definen en el estándar EIA[1], que para el RS232C son

- Un *espacio* (0 lógico) entre +3 y +25V
- Una *marca* (1 lógico) entre -3 y -25V
- Los valores entre +3V y -3V representan estados indefinidos
- Corriente de cortocircuito <500mA

(entre otros parámetros como: capacitancia máxima de línea, baudrate máximo, etc.)

Define una comunicación entre:

- DTE: Data Terminal Equipment (p.e. PC)
- DCE: Data Communication Equipment (p.e. módem)

[1] EIA: Electronics Industry Association - RS: Recommended Standard EIA-232C (1969), EIA-232D (1986), EIA-232 (1991)

# Hardware

Las especificaciones eléctricas del puerto serie se definen en el estándar EIA[1], que para el RS232C son

- Un *espacio* (0 lógico) entre +3 y +25V
- Una *marca* (1 lógico) entre -3 y -25V
- Los valores entre +3V y -3V representan estados indefinidos
- Corriente de cortocircuito <500mA

(entre otros parámetros como: capacitancia máxima de línea, baudrate máximo, etc.)

Define una comunicación entre:

- DTE: Data Terminal Equipment (p.e. PC)
- DCE: Data Communication Equipment (p.e. módem)

Es posible comunicar dos PCs (DTE) mediante cable null-model

[1] EIA: Electronics Industry Association - RS: Recommended Standard EIA-232C (1969), EIA-232D (1986), EIA-232 (1991)

# Hardware

Las especificaciones eléctricas del puerto serie se definen en el estándar EIA[1], que para el RS232C son

- Un *espacio* (0 lógico) entre +3 y +25V
- Una *marca* (1 lógico) entre -3 y -25V
- Los valores entre +3V y -3V representan estados indefinidos
- Corriente de cortocircuito <500mA

(entre otros parámetros como: capacitancia máxima de línea, baudrate máximo, etc.)

Define una comunicación entre:

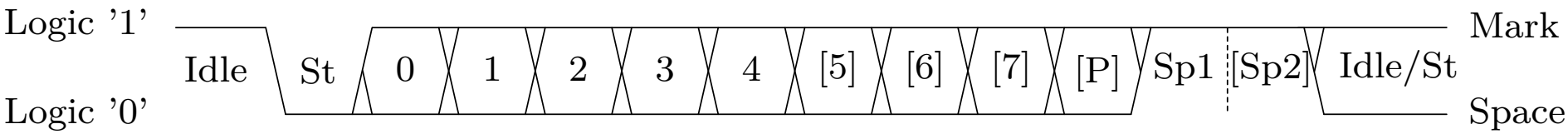
- DTE: Data Terminal Equipment (p.e. PC)
- DCE: Data Communication Equipment (p.e. módem)

Es posible comunicar dos PCs (DTE) mediante cable null-model

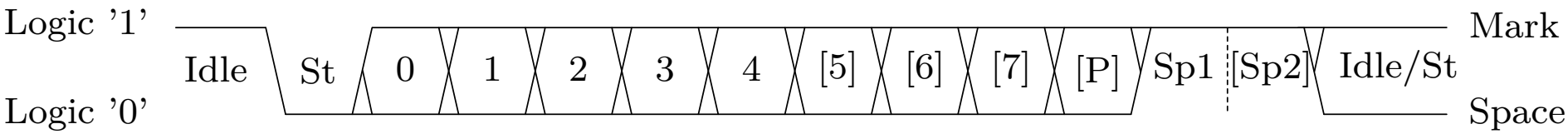
Conector DB-9/DB-25 macho

[1] EIA: Electronics Industry Association - RS: Recommended Standard EIA-232C (1969), EIA-232D (1986), EIA-232 (1991)

# Trama de comunicación

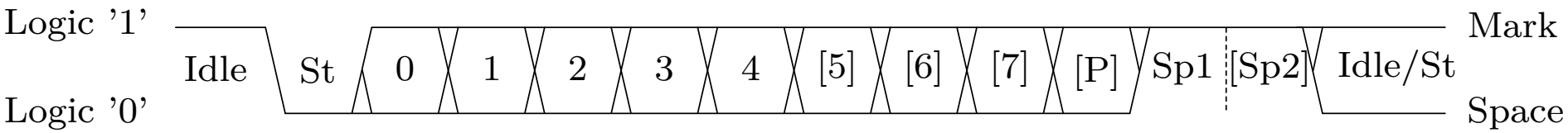


# Trama de comunicación



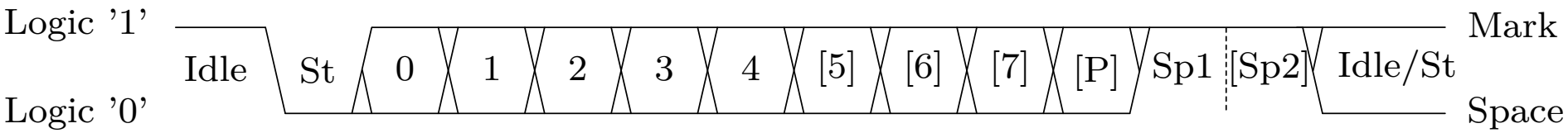

- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]

# Trama de comunicación



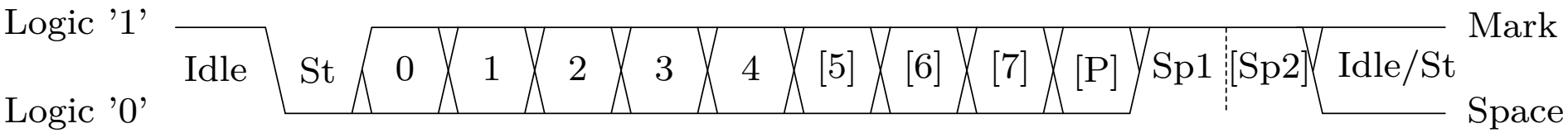

- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]

# Trama de comunicación



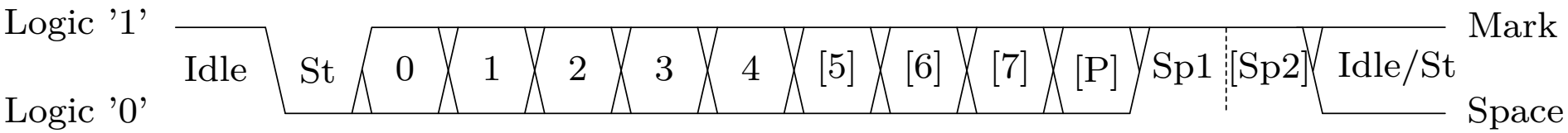

- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos

# Trama de comunicación



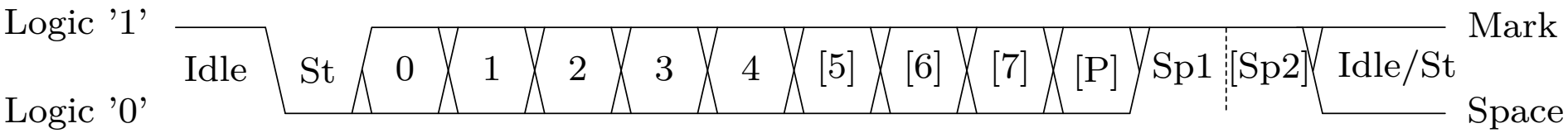

- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos
- P: Parity bit (bit de paridad)

# Trama de comunicación




- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos
- P: Parity bit (bit de paridad)
- Sp: Stop bit (bit de parada) [Logic 1]

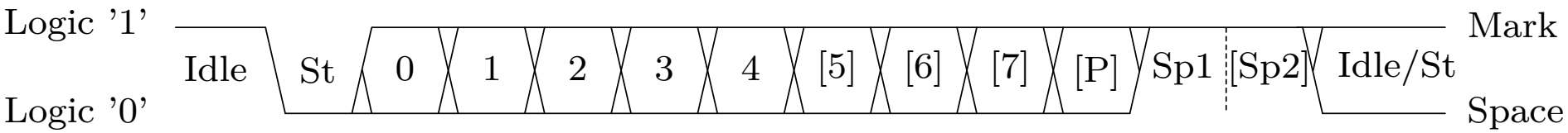
# Trama de comunicación




- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos
- P: Parity bit (bit de paridad)
- Sp: Stop bit (bit de parada) [Logic 1]

La comunicación es asíncrona (El bit menos significativo –LSb– se envía primero)

# Trama de comunicación



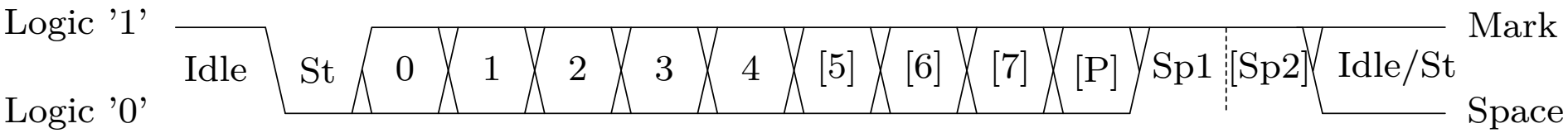

- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos
- P: Parity bit (bit de paridad)
- Sp: Stop bit (bit de parada) [Logic 1]

La comunicación es asíncrona (El bit menos significativo –LSb– se envía primero)

Algunas alternativas de tramas

- 8N1: 8 bits de datos, sin (N) bit de paridad, y 1 bit de stop.
- 5N2: 5 bits de datos, sin bit de paridad, y 2 bits de stop.

# Trama de comunicación




- Idle: No hay transferencia de datos (TxD y RxD) [Logic 1]
- St: Start bit (bit de arranque) [Logic 0]
- (n): Bits de datos
- P: Parity bit (bit de paridad)
- Sp: Stop bit (bit de parada) [Logic 1]

La comunicación es asíncrona (El bit menos significativo –LSb– se envía primero)

Algunas alternativas de tramas

- 8N1: 8 bits de datos, sin (N) bit de paridad, y 1 bit de stop.
- 5N2: 5 bits de datos, sin bit de paridad, y 2 bits de stop.

¿Qué duración tiene cada bit?

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | |
| 2 (3) | RD | Receive Data | |
| 7 (4) | RTS | Request To Send | |
| 8 (5) | CTS | Clear To Send | |
| 6 (6) | DSR | Data Set Ready | |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | |
| 7 (4) | RTS | Request To Send | |
| 8 (5) | CTS | Clear To Send | |
| 6 (6) | DSR | Data Set Ready | |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

TD: Salida de datos seriales, TxD

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | |
| 8 (5) | CTS | Clear To Send | |
| 6 (6) | DSR | Data Set Ready | |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

RD: Entrada de datos seriales, RxD

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | |
| 6 (6) | DSR | Data Set Ready | |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

RTS: Le indica al módem que la UART está lista para comunicarse

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | DTE ← DCE |
| 6 (6) | DSR | Data Set Ready | |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

CTS: Indica que el módem está listo para comunicarse

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | DTE ← DCE |
| 6 (6) | DSR | Data Set Ready | DTE ← DCE |
| 5 (7) | SG | Signal Ground | |
| 1 (8) | DCD | Data Carrier Detect | |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

DSR: Le indica a la UART que el módem está listo para establecer una conexión

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | DTE ← DCE |
| 6 (6) | DSR | Data Set Ready | DTE ← DCE |
| 5 (7) | SG | Signal Ground | DTE — DCE |
| 1 (8) | DCD | Data Carrier Detect | DTE ← DCE |
| 4 (20) | DTR | Data Tearminal Ready | |
| 9 (22) | RI | Ring Indicator | |

DCD: Indica que el módem detecta “portadora”

# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | DTE ← DCE |
| 6 (6) | DSR | Data Set Ready | DTE ← DCE |
| 5 (7) | SG | Signal Ground | DTE — DCE |
| 1 (8) | DCD | Data Carrier Detect | DTE ← DCE |
| 4 (20) | DTR | Data Tearminal Ready | DTE → DCE |
| 9 (22) | RI | Ring Indicator | |

DTR: Opuesto a DSR. Le indica al módem que la UART está lista para establecer conexión

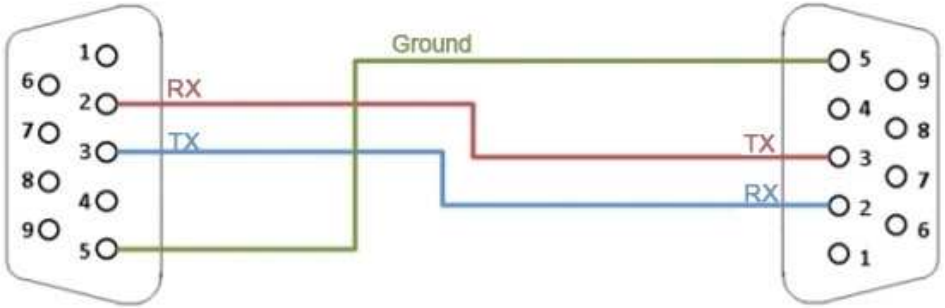
# Hardware – Conector y señales

| DB-9 (DB-25) Pin No. | Abbrev. | Full name | (PC) — (.) |
|---|---|---|---|
| 3 (2) | TD | Transmit Data | DTE → DCE |
| 2 (3) | RD | Receive Data | DTE ← DCE |
| 7 (4) | RTS | Request To Send | DTE → DCE |
| 8 (5) | CTS | Clear To Send | DTE ← DCE |
| 6 (6) | DSR | Data Set Ready | DTE ← DCE |
| 5 (7) | SG | Signal Ground | DTE — DCE |
| 1 (8) | DCD | Data Carrier Detect | DTE ← DCE |
| 4 (20) | DTR | Data Tearminal Ready | DTE → DCE |
| 9 (22) | RI | Ring Indicator | DTE ← DCE |

RI: Se activa ante la presencia de llamada

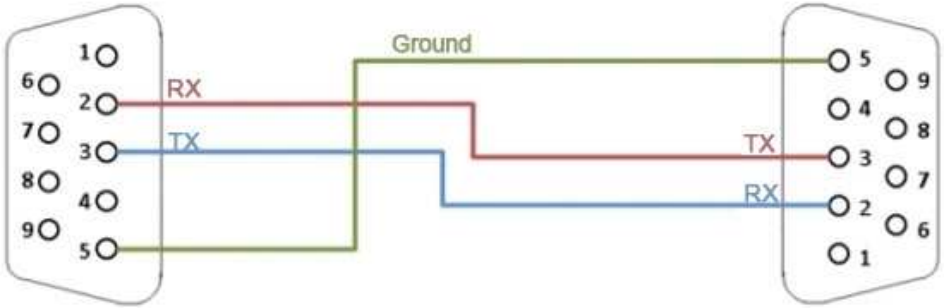
# Conexión null-módem

## Cable Null-Modem Simple

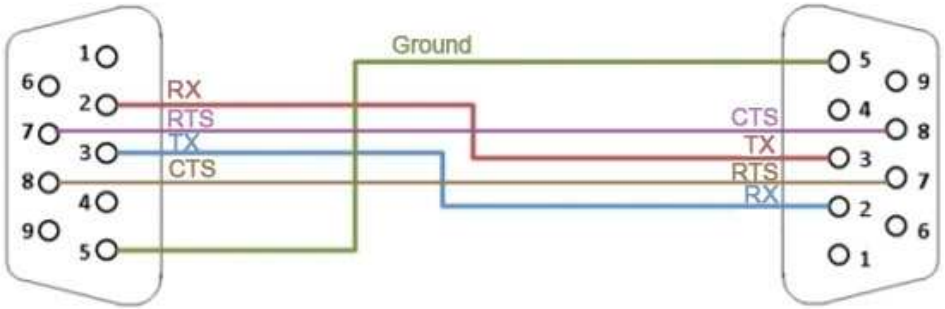


# Conexión null-módem

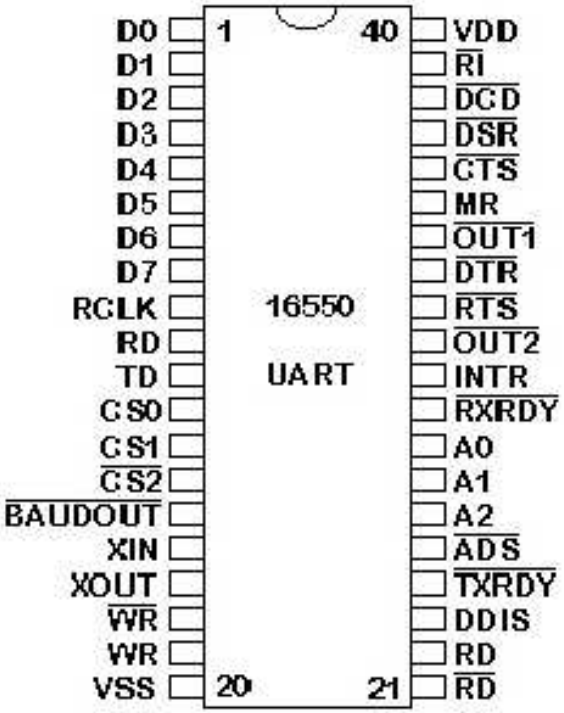
## Cable Null-Modem Simple

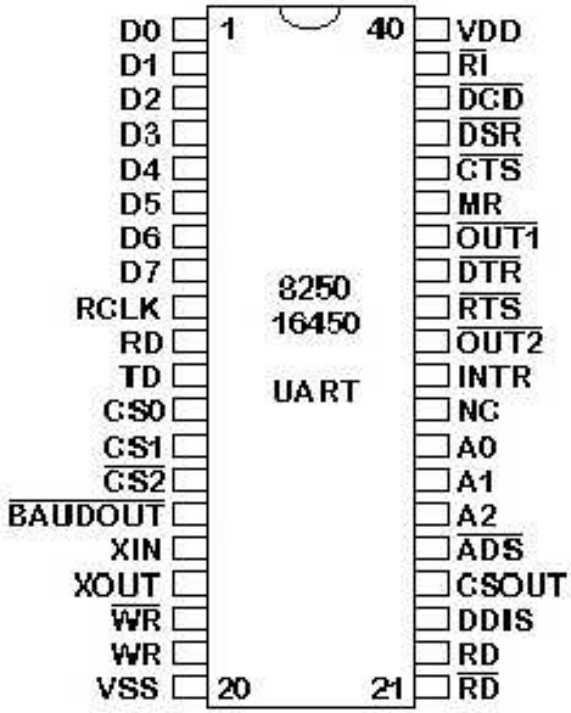



## Cable Null-Modem con Handshake

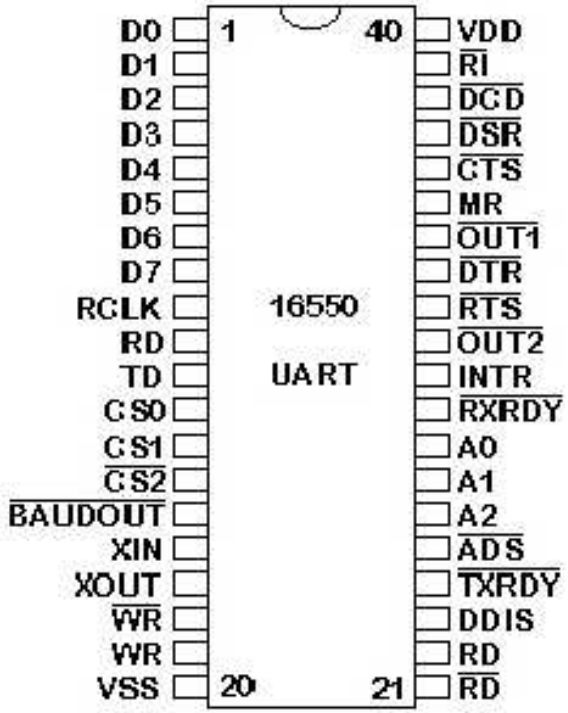


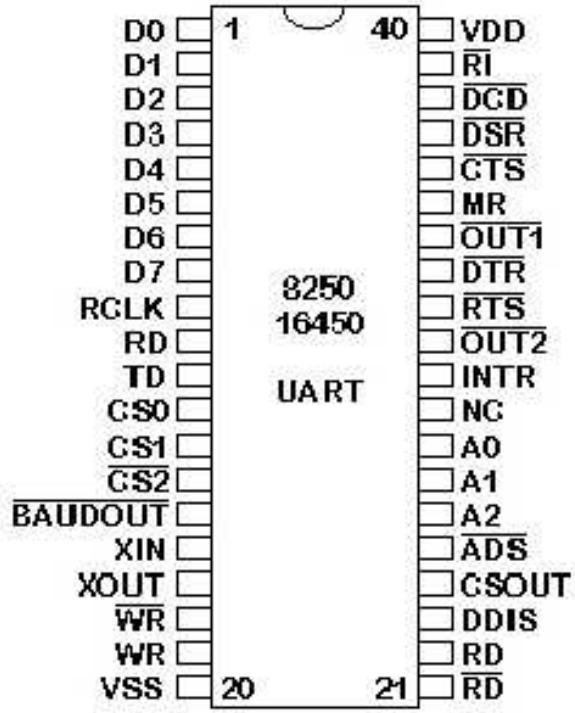
# Universidad Asynchronous Receive-Transmitter



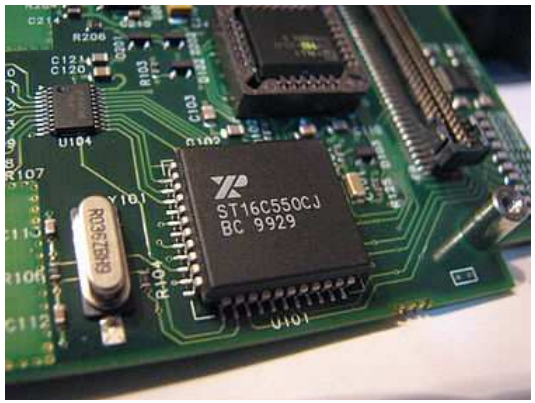



# Universidad Asynchronous Receive-Transmitter







# UART – Asignación de pines

| Nro. Pin | Nombre | Descripción |
|---|---|---|
| 1:8 | D0:D7 | Bus de datos |
| 9 | RCLK | Entrada de reloj ($f = 16$ baudrate) |
| 10 | RD | Recepción de datos |
| 11 | TD | Transmisión de datos |
| 12 | CS0 | Chip Select 0 |
| 13 | CS1 | Chip Select 1 |
| 14 | nCS2 | Chip Select 2 |
| 15 | nBAUDOUT | Salida BaudRate programable |
| 16 | XIN | Entrada cristal externo |
| 17 | XOUT | Salida cristal externo |
| 18 | nWR | Línea de escritura (invertida) |
| 19 | WR | Línea de escritura |
| 20 | VSS | Masa general |
| 21 | RD | Línea de entrada |
| 22 | nRD | Línea de entrada (invertida) |
| 23 | DDIS | Driver Disable |
| 24 | nTXRDY | Transmisor listo |
| 25 | nADS | Address Strobe |
| 26:28 | A0:A2 | Bits de dirección (000 – 111) |

# UART – Asignación de pines

| Nro. Pin | Nombre | Notas |
|---|---|---|
| 29 | nRXRDY | Receptor listo |
| 30 | INTR | Salida Interrupción |
| 31 | nOUT2 | Salida de usuario 2 |
| 32 | nRTS | Request To Send |
| 33 | nDTR | Data Terminal Ready |
| 34 | nOUT1 | Salida de usuario 1 |
| 35 | MR | Master Reset |
| 36 | nCTS | Clear To Send |
| 37 | nDSR | Data Set Ready |
| 38 | nDCD | Data Carrier Detect |
| 39 | nRI | Ring Indicator |
| 40 | VDD | Alimentación (+5V) |

# Dirección de puertos y registros

## Direcciones de los puertos

- 0x3F8: Puerto serie 1 (COM1)
- 0x2F8: Puerto serie 2 (COM2)
- 0x3E8: Puerto serie 3 (COM3)
- 0x2E8: Puerto serie 4 (COM4)

# Dirección de puertos y registros

## Direcciones de los puertos

- 0x3F8: Puerto serie 1 (COM1)
- 0x2F8: Puerto serie 2 (COM2)
- 0x3E8: Puerto serie 3 (COM3)
- 0x2E8: Puerto serie 4 (COM4)

Los **Registros** van desde BASE+0 hasta BASE+7

| BASE+ | DLAB | R/W | Arb. | Nombre |
|---|---|---|---|---|
| +0 | 0 | W | - | Transmitter Holding Buffer |
| | 0 | R | - | Receiver Buffer |
| | 1 | R/W | - | Divisor Latch Low Byte (DLLB) |
| +1 | 0 | R/W | IER | Interrupt Enable Register |
| | 1 | R/W | - | Divisor Latch High Byte (DLHB) |
| +2 | - | R | IIR | Interrupt Identification Register |
| | - | W | FCR | FIFO Control Register |
| +3 | - | R/W | LCR | Line Control Register |
| +4 | - | R/W | MCR | Modem Control Register |
| +5 | - | R | LSR | Line Status Register |
| +6 | - | R | MSR | Modem Status Register |
| +7 | - | R/W | - | Scratch Register |

# El bit DLAB y la velocidad de comunicación

DLAB: Division Latch Access Bit

- Permite tener 12 registros en 8 direcciones
- Cuando DBAL=1 a través del Line Control Register se tiene acceso a dos registros que permite fijar la velocidad de comunicación (bps)

# El bit DLAB y la velocidad de comunicación

DLAB: Division Latch Access Bit

- Permite tener 12 registros en 8 direcciones
- Cuando DBAL=1 a través del Line Control Register se tiene acceso a dos registros que permite fijar la velocidad de comunicación (bps)

La UART (oscilador de 1.8432MHz)

- Realiza una división por 16 $\Longrightarrow$ velocidad máxima de 115.200Hz (115.200bps)
- Para velocidades menores cuenta con un Generador de Baud Rate Programable controlado por dos registros

# El bit DLAB y la velocidad de comunicación

DLAB: Division Latch Access Bit

- Permite tener 12 registros en 8 direcciones
- Cuando DBAL=1 a través del Line Control Register se tiene acceso a dos registros que permite fijar la velocidad de comunicación (bps)

La UART (oscilador de 1.8432MHz)

- Realiza una división por 16 $\Longrightarrow$ velocidad máxima de 115.200Hz (115.200bps)
- Para velocidades menores cuenta con un Generador de Baud Rate Programable controlado por dos registros

| **Velocidad (BPS)** | **Divisor (dec)** | **DLHB** | **DLLB** |
|---|---|---|---|
| 50 | 2304 | 0x09 | 0x00 |
| 300 | 384 | 0x01 | 0x80 |
| 600 | 192 | 0x00 | 0xC0 |
| 2400 | 48 | 0x00 | 0x30 |
| 4800 | 24 | 0x00 | 0x18 |
| 9600 | 12 | 0x00 | 0x0C |
| 19200 | 6 | 0x00 | 0x06 |
| 38400 | 3 | 0x00 | 0x03 |
| 57600 | 2 | 0x00 | 0x02 |
| 115200 | 1 | 0x00 | 0x01 |

# Registros – Line Control Register (LCR)

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| i | DLAB | PS2 | PS1 | PS0 | LSB | WL1 | WL0 |

- DLAB: Divisor Latch Access Bit
- SBE: Set Break Enable
- PS: Parity Select
  - xx0: No parity
  - 001: Odd parity
  - 011: Even parity
  - 101: High parity
  - 111: Low parity
- LSB: Lenght of Stop Bit (0: 1 stop bit, 1: 2 stop bit)
- WL: Word length
  - 00: 5 bits
  - 01: 6 bits
  - 10: 7 bits
  - 11: 8 bits

# Registros – Otros registros

La descripción de los demás registro se puede ver el los documentos de bibliografía

# Puertos series en GNU/Linux

- Los archivos de dispositivos para los puertos serie en Linux son: `/dev/ttyS0`, `/dev/ttyS1`, `/dev/ttyUSB0`, `/dev/ttyACM0`, etc.

# Puertos series en GNU/Linux

- Los archivos de dispositivos para los puertos serie en Linux son: `/dev/ttyS0`, `/dev/ttyS1`, `/dev/ttyUSB0`, `/dev/ttyACM0`, etc.
- Para ver el archivo de dispositivo creado al conectar el hardware (Arduino o adaptador USB-serie) utilizando el comando `dmesg`

# Puertos series en GNU/Linux

- Los archivos de dispositivos para los puertos serie en Linux son: `/dev/ttyS0`, `/dev/ttyS1`, `/dev/ttyUSB0`, `/dev/ttyACM0`, etc.
- Para ver el archivo de dispositivo creado al conectar el hardware (Arduino o adaptador USB-serie) utilizando el comando `dmesg`

Ejemplo de salida de `dmesg` con placa Arduino UNO

```
usb 3-1: new full-speed USB device number 15 using xhci_hcd
usb 3-1: New USB device found, idVendor=1a86, idProduct=7523
usb 3-1: New USB device strings: Mfr=0, Product=2, SerialNumber=0
usb 3-1: Product: USB2.0-Serial
ch341 3-1:1.0: ch341-uart converter detected
usb 3-1: ch341-uart converter now attached to ttyUSB0
```

# Puertos series en GNU/Linux

- Los archivos de dispositivos para los puertos serie en Linux son: `/dev/ttyS0`, `/dev/ttyS1`, `/dev/ttyUSB0`, `/dev/ttyACM0`, etc.
- Para ver el archivo de dispositivo creado al conectar el hardware (Arduino o adaptador USB-serie) utilizando el comando `dmesg`

Ejemplo de salida de `dmesg` con placa Arduino UNO

```
usb 3-1: new full-speed USB device number 15 using xhci_hcd
usb 3-1: New USB device found, idVendor=1a86, idProduct=7523
usb 3-1: New USB device strings: Mfr=0, Product=2, SerialNumber=0
usb 3-1: Product: USB2.0-Serial
ch341 3-1:1.0: ch341-uart converter detected
usb 3-1: ch341-uart converter now attached to ttyUSB0
```

Ejemplo de salida de `dmesg` con adaptador USB-serie

```
usb 3-2: new full-speed USB device number 17 using xhci_hcd
usb 3-2: New USB device found, idVendor=067b, idProduct=2303
usb 3-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
usb 3-2: Product: USB-Serial Controller
usb 3-2: Manufacturer: Prolific Technology Inc.
pl2303 3-2:1.0: pl2303 converter detected
usb 3-2: pl2303 converter now attached to ttyUSB1
```

# Puertos series en GNU/Linux

- Para modificar y ver la configuración actual de un puerto serie (archivo de dispositivo en `/dev`) se puede utilizar el comando `stty`
  - Para ver la velocidad de comunicación actual

```
> stty -F /dev/ttyUSB0
speed 9600 baud; line = 0;
```

  - Para modificar la velocidad de comunicación

```
> stty -F /dev/ttyUSB1 115200
> stty -F /dev/ttyUSB1
speed 115200 baud; line = 0;
```

# Puertos series en GNU/Linux

- Para modificar y ver la configuración actual de un puerto serie (archivo de dispositivo en `/dev`) se puede utilizar el comando `stty`
  - Para ver la velocidad de comunicación actual

```
> stty -F /dev/ttyUSB0
speed 9600 baud; line = 0;
```

  - Para modificar la velocidad de comunicación

```
> stty -F /dev/ttyUSB1 115200
> stty -F /dev/ttyUSB1
speed 115200 baud; line = 0;
```

- Ejemplo con Arduino:
  - Abrir IDE Arduino y grabar el ejemplo `AnalogReadSerial`
  - Ver valores enviados desde el Monitor Serial (IDE)
  - Ver valores enviados desde `cutecom`

# Puertos series en GNU/Linux

- Para modificar y ver la configuración actual de un puerto serie (archivo de dispositivo en `/dev`) se puede utilizar el comando `stty`
  - Para ver la velocidad de comunicación actual

```
> stty -F /dev/ttyUSB0
speed 9600 baud; line = 0;
```

  - Para modificar la velocidad de comunicación

```
> stty -F /dev/ttyUSB1 115200
> stty -F /dev/ttyUSB1
speed 115200 baud; line = 0;
```

- Ejemplo con Arduino:
  - Abrir IDE Arduino y grabar el ejemplo `AnalogReadSerial`
  - Ver valores enviados desde el Monitor Serial (IDE)
  - Ver valores enviados desde `cutecom`
- Algunas terminales: `screen`, `cutecom`, etc.

# Puertos series en GNU/Linux

# Puertos series en GNU/Linux

- Uso de `socat` para generación de puertos "virtuales"
  ```
  > socat PTY,link=/tmp/ttyS0 PTY,link=/tmp/ttyS1
  ```
- Mostar en terminal lo recibido por el puerto serie
  ```
  > cat /tmp/ttyS1
  ```
- Enviar una cadeba a un puerto serie
  ```
  > cat /tmp/ttyS0
  ```
- Enviar una cadena desde `cutecom`

# Programación – Abrir y cerrar el puerto serie

```c
#include <stdio.h> /* Standard input/output definitions */
#include <fcntl.h> /* File control definitions */
#include <unistd.h> /* UNIX standard function definitions */

int main(void) {
  int fd; /* File descriptor for the port */

  fd = open("/dev/ttyUSB0", O_RDWR | O_NOCTTY | O_NDELAY);

  if(fd == -1)
     /* ERROR */

  close(fd);
  return 0;
}
```

Flags:

- `O_RDWR`: Lectura/Escritura
- `O_NOCTTY`: Para que no sea una terminal de control (?)
- `O_NDELAY`: Ignora la señal DCD (sino, el proceso se duerme hasta activarse DCD)

# Programación – Escribir y leer datos

```c
#include <stdio.h> /* Standard input/output definitions */
#include <fcntl.h> /* File control definitions */
#include <unistd.h> /* UNIX standard function definitions */

int main(void) {
  int n, fd;
  char* data;

  fd = open("/dev/ttyUSB0", O_RDWR | O_NOCTTY | O_NDELAY);
  if(fd == -1) {} /* ERROR */

  /* Writing data to the port */
  n = write(fd, "Hello\n", 6);
  if(n < 0)
    /* ERROR */

  /* Reading data from the port */
  /* n = read(fid, data, 4); */
  /* if(n < 4) */

  close(fd);
  return 0;
}
```

# Configuración del puerto serie (termios)

Terminales en sistemas POSIX (Portable Operating System Interface (UNIX))

- `termios.h`: estructura de control de terminal y funciones de control
- Cambiar parámetros tales como velocidad de comunicación, tamaño trama, etc.

Las dos funciones más importantes son: `tcgetattr()`, `tcsetattr()`

Miembros de la estructura termios:

- `c_cflags`: Opciones de control
- `c_lflags`: Opciones de línea
- `c_iflags`: Opciones de entrada
- `c_oflags`: Opciones de salida
- `c_cc`: Caracteres de control
- `c_ispeed`: Baudrate de entrada (interfaz nueva)
- `c_ospeed`: Baudrate de salida (interfaz nueva)

# Configuración del puerto serie (termios)

```
struct termios options;
/* Get the current options for the port */
tcgetattr(fd, &options);

/* Set the baud rates to 19200 */
cfsetispeed(&options, B19200);
cfsetospeed(&options, B19200);

/* Enable the receiver and set local mode */
options.c_cflag |= (CLOCAL | CREAD);

/* No parity 8N1 */
options.c_cflag &= ~PARENB; /* No parity */
options.c_cflag &= ~CSTOPB; /* 1 stop bit */
options.c_cflag &= ~CSIZE; /* Mask the character size bits */
options.c_cflag |= CS8;

/* Set the new options for the port */
tcsetattr(fd, TCSANOW, &options);
```

# Bibliografía


- *Interfacing the Serial / RS232 Port*, V5.0 (web)
- *Serial Programming Guide for POSIX Operating Systemsr*, 5th Edition–3rd Revision. (web)
- *Beginning Linux Programming*, 4th Edition, Neil Matthew, Richard Stones (ISBN: 978-0-470-14762-7) (Libro)
- *GNU C Library –Terminal modes– (link)*